PREDICTING POWER OUTAGES USING GRAPH NEURAL NETWORKS

Damian Owerko, Fernando Gama and Alejandro Ribeiro E-mail: owerko@sas.upenn.edu
Dept. of Electrical & Systems Engineering, University of Pennsylvania

Predicting Power Outages

- Power outages have huge economic cost
  - ⇒ \$22B and \$135B annually [Campbell '12]
  - ⇒ Most caused by weather conditions [Panteli '15]
- Weather data can be expressed as a graph signal
  - ⇒ Graph based on distance between weather stations

Objectives
- Predict power outages using weather data
- Validate efficacy of graph neural networks

Convolutional Neural Networks for Graphs

- Existing CNNs ⇒ Remarkable performance in processing regular data
  - ⇒ Convolution, pooling need a regular, multi-resolution domain
- Lots of data presents alternative irregular structural information
  - ⇒ Especially, many problems in wireless systems (network ⇔ graph)

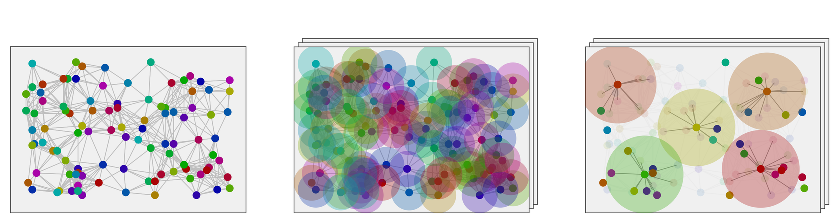
Graph Neural Networks (GNNs) generalize CNNs
⇒ Convolution ⇒ Linear shift-invariant graph filters
⇒ Pooling ⇒ Local nonlinearity followed by downsampling

Convolutional Neural Networks on Graphs

- Network structure ⇒ Graph matrix $\mathbf{S}$ (Adjacency $\mathbf{A}$, Laplacian $\mathbf{L}$)
  - ⇒ $[\mathbf{S}]_{ij}$ = Relationship between i and j (underlying graph support)
- Define a signal $\mathbf{x}$ on top of the graph
  - ⇒ $[\mathbf{x}]_i$ = Signal value at node i
- Graph Signal Processing ⇒ Exploit structure encoded in $\mathbf{S}$ to process $\mathbf{x}$
- Generate features through (local) convolution and (local) pooling
  - ⇒ Convolution is a linear shift invariant filter

$$\mathbf{y} = h_0\mathbf{S}^0\mathbf{x} + h_1\mathbf{S}^1\mathbf{x} + h_2\mathbf{S}^2\mathbf{x} + \ldots + h_{K-1}\mathbf{S}^{K-1}\mathbf{x} = \sum_{k=0}^{K-1} h_k\mathbf{S}^k\mathbf{x} := \mathbf{H}\mathbf{x}$$

⇒ Pooling summarizes information in graph neighborhoods



- But beyond first layer there's no graph connecting signal components
- Define pooling operations on $\mathbf{S}$ as well
  - ⇒ Clustering [Defferrard '16]
  - ⇒ Selection [Gama '18]
  - ⇒ Aggregation [Gama '18]

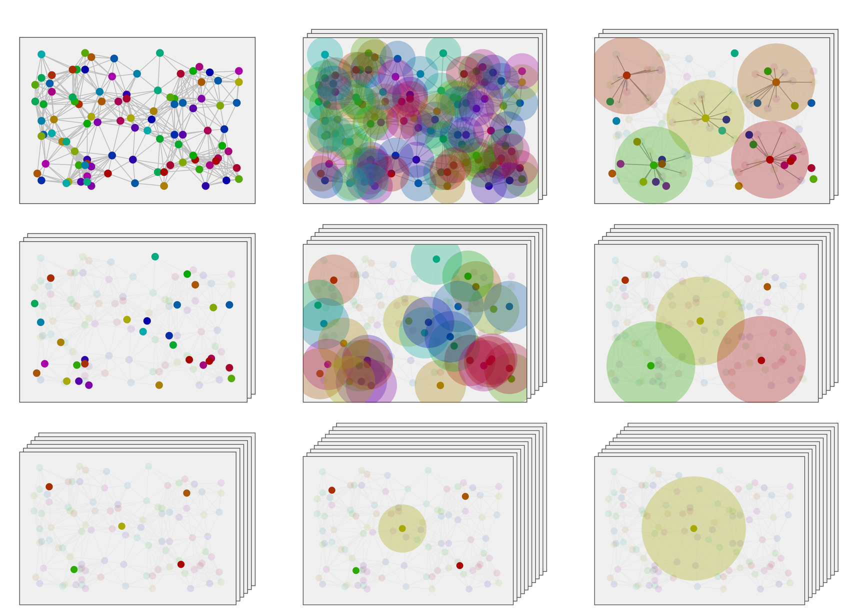
Graph Neural Network Architectures

Clustering
- Multi-scale hierarchical clustering algorithm applied at each layer
- Corresponding $\mathbf{S}$ has a lower dimension
- More details can be found in [Defferrard '16]

Selection
- Linear shift invariant graph filters to build convolutional features
- Pooling as subsampling ⇒ Remember sampled locations on graph
- Use zero padding for convolutional features at hidden layers

Aggregation
- Successively apply graph shift. Store observed values at one node
- Creates signal with time structure that incorporates graph topology
- We can now perform convolution and pooling on the time domain
- Multi-node version with outer layers using zero padding

Selection GNN



Selection GNNs

- Bypass the need to generate new graphs
  - ⇒ Achieved by downsampling and zero padding.
- Selection defined by matrix $\mathbf{C}_\ell \in \{0,1\}^{N_\ell \times N_{\ell-1}}$
- Feature $\mathbf{x}_1^g$ is not supported on the same graph (smaller dimension)
  - ⇒ Problem can be solved by remembering location of sampled nodes
- Place signal $\mathbf{x}_{\ell-1}^g$ on the original input graph
  - ⇒ Zero-pad input features

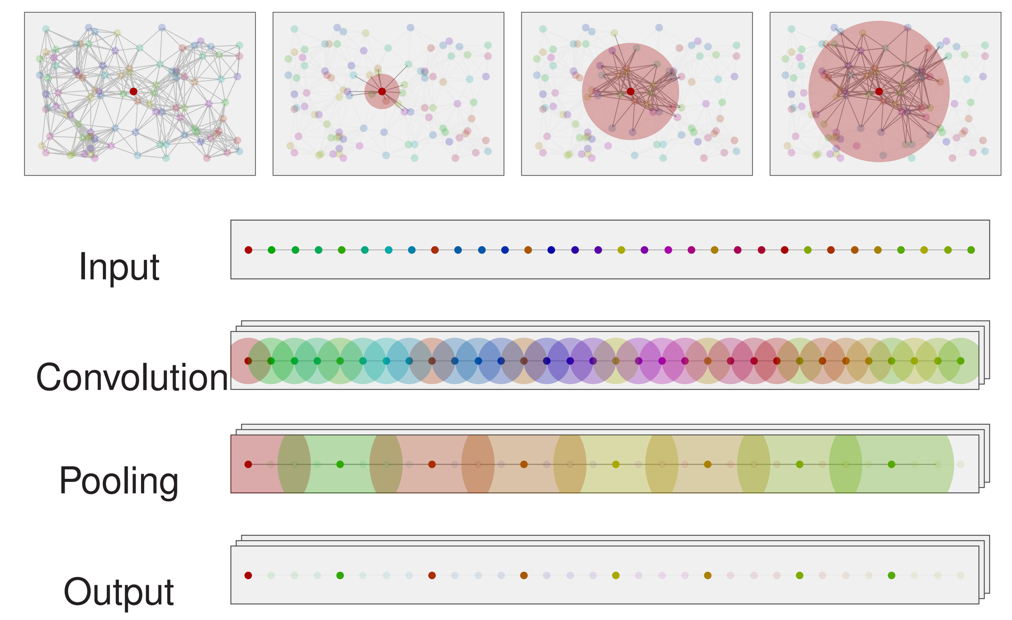
$$\tilde{\mathbf{x}}_{\ell-1}^g = \mathbf{D}_{\ell-1}^{\mathsf{T}}\mathbf{x}_{\ell-1}^g,\ \mathbf{D}_{\ell-1} = \mathbf{C}_\ell\mathbf{C}_{\ell-1}\cdots\mathbf{C}_1 \in \{0,1\}^{N_\ell \times N}$$

- Therefore the convolution operation becomes

$$\mathbf{H}_\ell^{fg}\mathbf{x}_{\ell-1}^g = \mathbf{D}_{\ell-1}\left(\sum_{k=0}^{K_\ell-1}[\mathbf{h}_\ell^{fg}]_k\mathbf{S}^k\right)\mathbf{D}_{\ell-1}^{\mathsf{T}}\mathbf{x}_{\ell-1}^g$$

- "Graph filter" using sampled k-shift matrices
  - ⇒ $\mathbf{S}_\ell^{(k)} := \mathbf{D}_{\ell-1}\mathbf{S}^k\mathbf{D}_{\ell-1}^{\mathsf{T}}$
- ***Pooling.*** Get α_ℓ-hop neighborhood using $\mathbf{S}_\ell^{(k)}$ for some $k \le \alpha_\ell$

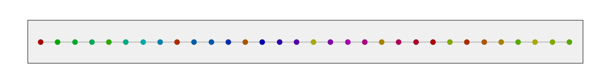
Aggregation GNN



Regular Convolution in Aggregation GNNs

- Input $\mathbf{x}_0^g$ is a signal over known N-node graph
- Select node $p \in \mathcal{V}$ ⇒ Perform N local exchanges
- Consecutive elements encode nearby neighbors

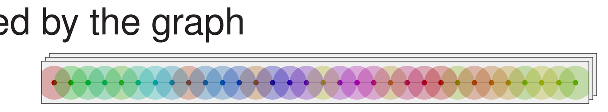
$$\mathbf{z}^g(p,N) = \left[[\mathbf{x}_0^g]_p, [\mathbf{S}\mathbf{x}_0^g]_p, [\mathbf{S}^2\mathbf{x}_0^g]_p, \ldots, [\mathbf{S}^{N-1}\mathbf{x}_0^g]_p\right]^{\mathsf{T}}$$



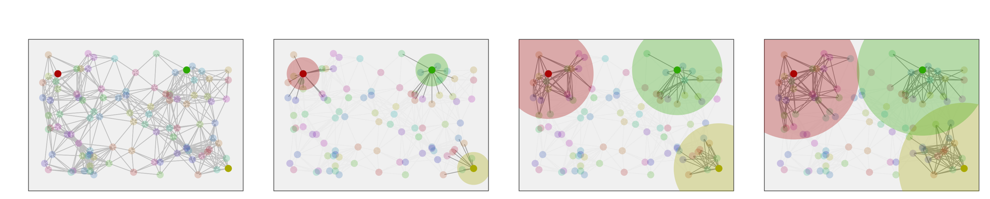
- This resulting signal has a regular structure
  - ⇒ We can use a regular convolution

$$\left[\mathbf{u}_1^{fg}\right]_n = \left[\mathbf{h}_1^{fg} * \mathbf{z}_p\right]_n = \sum_{k=0}^{K_1-1}\left[\mathbf{h}_1^{fg}\right]_k\left[\mathbf{z}_p\right]_{n-k} = \sum_{k=0}^{K_1-1}\left[\mathbf{h}_1^{fg}\right]_k\left[\mathbf{S}^{n-k}\mathbf{x}_0^g\right]_p$$

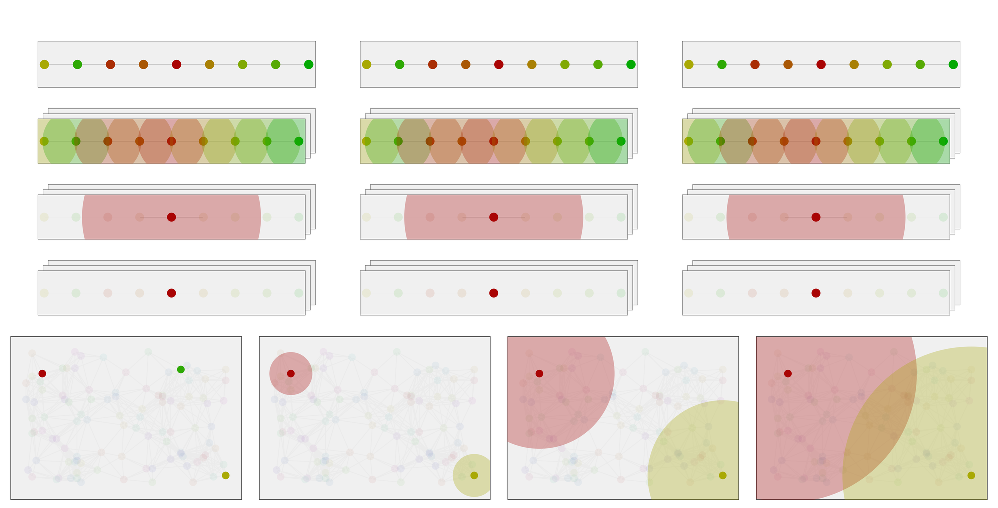
⇒ Effectively relates neighboring information encoded by the graph



- Therefore regular pooling and downsampling can be used as well
- N exchanges can be expensive ⇒ select a subset of nodes $\mathcal{P} \subset \mathcal{V}$
  - ⇒ This leads to multi-node aggregation, a hybrid between selection and aggregation

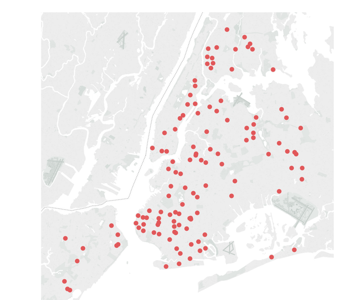


Multi-Node Aggregation GNNs



Dataset

- We use weather data to predict power outage events
- Considering Jan '11 - Dec '15
- NYC Weather data
  - ⇒ Hourly logs from 123 weather stations
  - ⇒ 26,304 datapoints
- NY electrical disturbance events
  - ⇒ 25 such events occur
- Preprocessing needed to collate datasets



- Some weather data was missing from the original dataset
  - ⇒ Implement a greedy algorithm to select $N = 25$ stations
  - ⇒ and a subset of datapoints with no missing datapoints
- In total there are 5,777 datapoints (hours)
  - ⇒ during 218 of which there was a major disturbance event
- 10% of the dataset is randomly chosen as a test set
  - ⇒ the rest is used for training and validation

Defining a Graph Signal

- Each weather station takes a variety of hourly measurements
  - ⇒ we use pressure, temperature, wind speed, pressure rate per hour, humidity, humidity rate per hour and precipitation rate
- At each hour we consider a datapoint $(\mathbf{x}^g, y)$
  - ⇒ $y = 1$ if major electrical disturbance at that hour, otherwise $y = 0$
  - ⇒ $[\mathbf{x}^g]_i$ is the weather feature g at station i

Define the graph shift operator $\mathbf{S}$ based on distance between stations
- We apply a Gaussian kernel to $d(i,j)$ ⇒ distance between stations i, j

$$\mathbf{S}_{ij} = \exp\left(-\frac{d(i,j)^2}{2\sigma^2}\right) \quad \text{where } \sigma = 0.1$$

- Additionally a threshold, ϵ_w, is applied to $\mathbf{S}$

Selected References

- Earth Networks, "Historical observations," 13 March 2018.
- M. Defferrard et al., "Convolutional neural networks on graphs with fast localized spectral filtering," in Neural Inform. Process. Syst. 2016. Barcelona, Spain: NIPS Foundation, 5-10 Dec. 2016..
- F. Gama et al., "Convolutional neural network architectures for signals supported on graphs," arXiv:1805.00165v1 [eess.SP], 1 May 2018.
- M. Panteli and P. Mancarella, "Influence of extreme weather and climate change on the resilience of power systems: Impacts and possible mitigation strategies," Electric Power Systems Research, vol. 127, pp. 259?270, Oct. 2015.
- R. J. Campbell, "Weather-related power outages and electric system resiliency," Congressional Research Service, CRS Report for Congress 7-5700 (R42696), 28 Aug. 2012.
- U.S. Energy Information Administration, "Electric power monthly," 25 March 2014.

Numerical Experiments

We compare the performance of the architectures against several baseline methods.

Neural network
- We concatenate all features into one long vector of length $N \times 7$
- Perform a hyperparameter search
  - ⇒ L FC layers, F_ℓ hidden units at layer ℓ and dropout d
- Due to prevalence of negative labels in the dataset NN converges to trivial solution
  - ⇒ The network would output 0 for all inputs
  - ⇒ We correct this by weighing positive labels more heavily $10:1$

Affine space model (PCA)
- We found using just pressure data results in highest performance
- Estimate interclass mean μ_y and covariance matrix $\mathbf{\Sigma}_y$
- We minimize the projection of the input on the class eigenvector matrix

$$\hat{y} = \underset{y \in \{0,1\}}{\operatorname{argmin}} ||\mathbf{V}_y(\mathbf{x} - \mu_y)||$$

Architecture	Accuracy	F1 Score
Clustering	68%	2.42
Selection	48%	2.60
Aggregation	65%	2.25
Multi-Node	74%	1.90
No pooling	86%	1.04
Neural Network	61%	2.88
PCA	86%	1.04

Results

- We predict power outages in NY from weather data
- This is used to compare graph neural networks against baseline methods
  - ⇒ a neural network and an affine space model
- No pooling yielded the best results with the highest F1 score
- However, all GNN architectures outperform the baseline methods

Conclusions and Future Work

- Graph neural networks outperform NN at this task
  - ⇒ Simultaneously, they have far fewer parameters
- All architectures managed to perform better than the trivial solution
- The best performing architecture brings a 70% improvement in prediction error
- Successfully demonstrated the use of GNNs
- Other potential architectures such as Graph RNNs
  - ⇒ Applications in machine translation